2019年　C言語講座

~第3回　DO文,WHILE文編~

目次

目次

1. はじめに

- どんな時にdo文,while文,for文を使うのか？

・処理を繰り返したい時に使う

例）

・数字の１～１００までを端末に表示

・入力した数字から１ずつ減算して,カウントダウンを表示する

printfを100回使う…？
入力した値によって変わる…

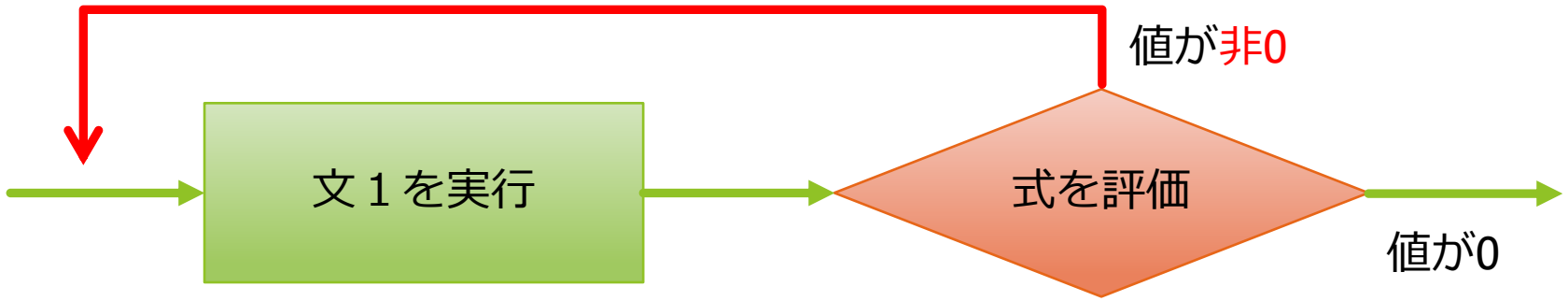
do文などを使えばもっと簡潔にかける！！

目次

２．　do文

- do文の書き方

do 文１ while(式);



・まず文１が実行される,その後()内の式を評価して非０なら
もう一度文１が実行される。値が０ならループから抜ける。

＊繰り返しのことをループという。繰り返しの対象（文１）を
ループ本体という。

2． do文

- do文の書き方（練習）

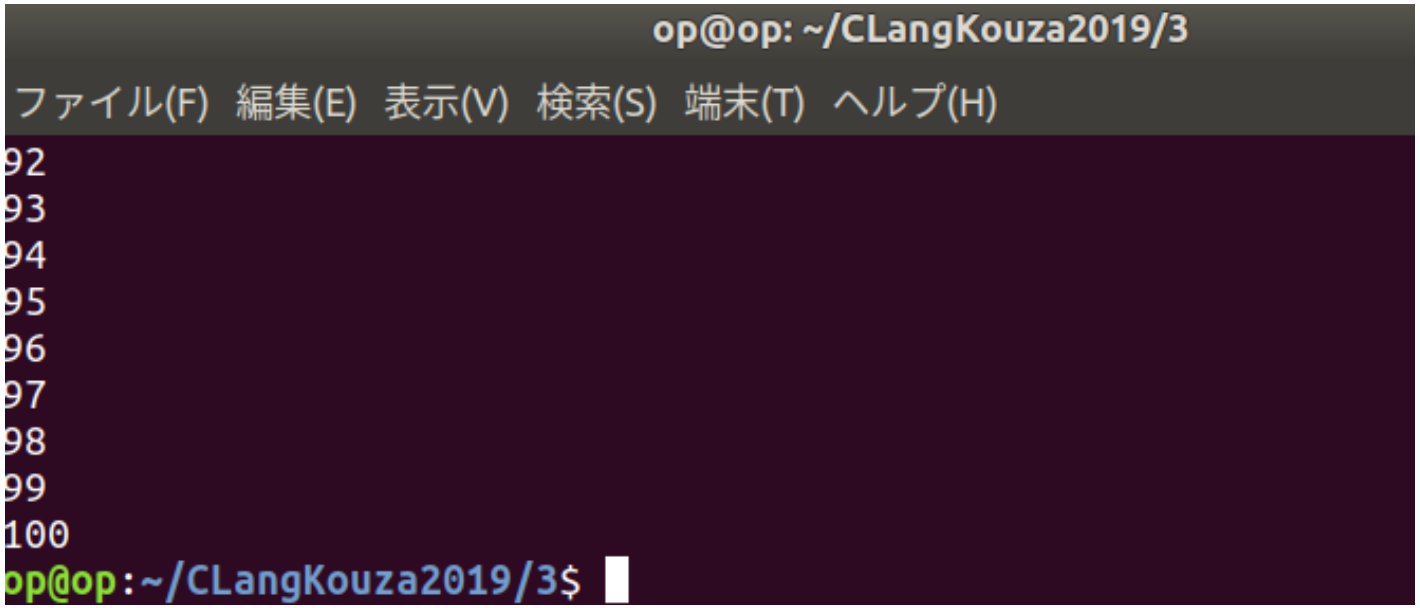
練習：
例）数字の１から100までを端末に表示

```
/* loop_do.c */
#include <stdio.h>

int main(void)
{
	int no = 0;

	do{
		no = no + 1;
		printf(“%d¥n”, no);

	}while(no <  100);

	return 0;
}
```

2． do文

- do文の書き方（練習）

・実行結果

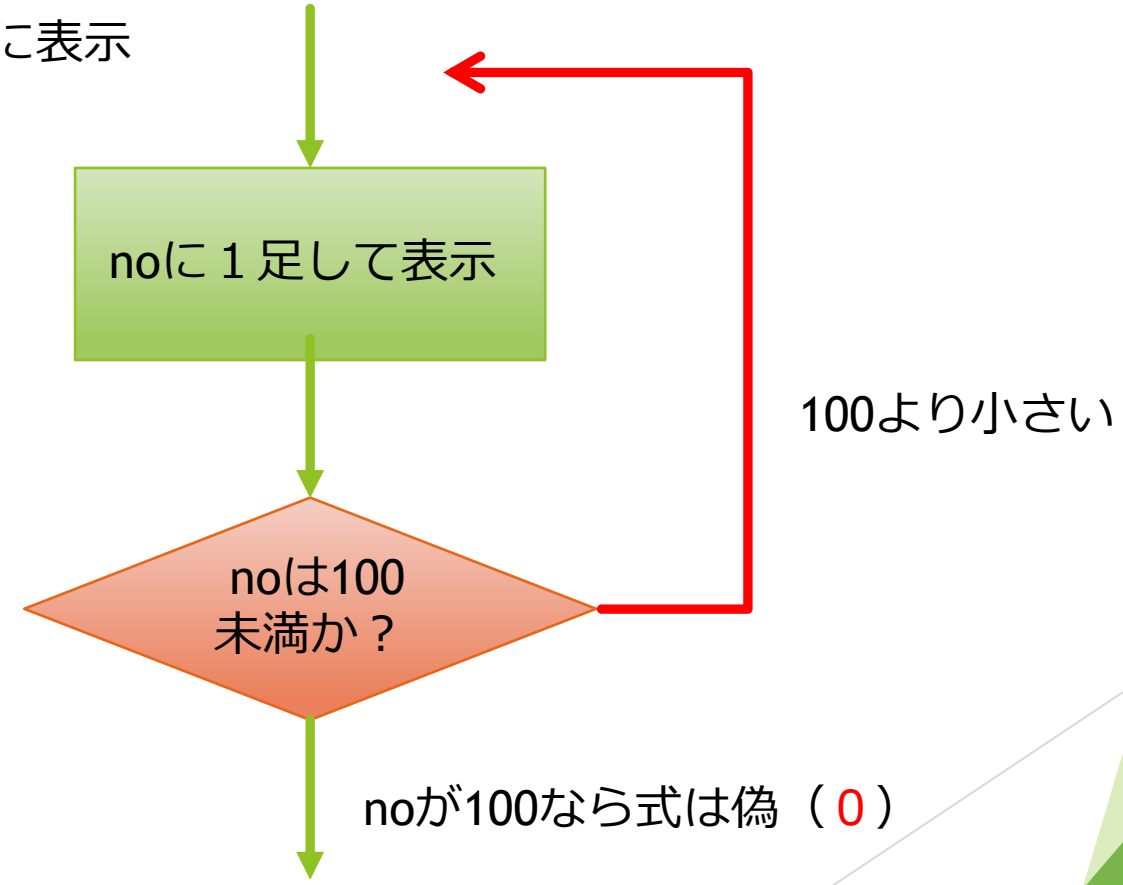


2． do文

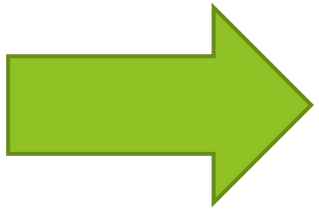
- do文の書き方（解説）

練習：
例）数字の１から100までを端末に表示

```
/* loop_do.c */
#include <stdio.h>

int main(void)
{
	int no = 0;

	do{
		no = no + 1;
		printf(“%d¥n”, no);

	}while(no <  100);

	return 0;
}
```



現状,while文内の式はループを繰り返すための継続条件...



式をループが終了する条件の否定に変更する

２．　do文

- 論理否定演算子とド・モルガンの法則

・論理否定演算子
　!a　　：aが０であれば１,そうでなければ０

・ド・モルガンの法則：
　“各条件の否定をとって、論理積・論理和を入れ替えた式”の否定はもとの条件と同じになる。

例）
　x && y　と !(!x || !y)は等しい

＊繰り返しのことをループという。繰り返しの対象（文１）をループ本体という。

2． do文

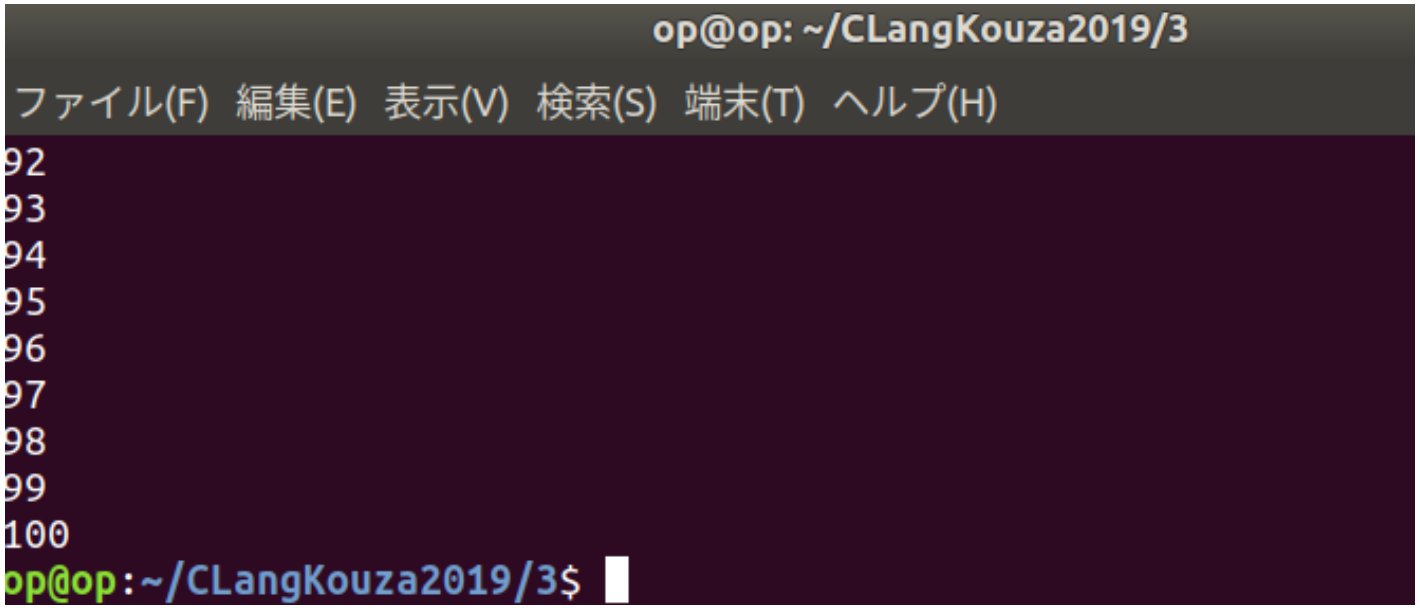
- 論理否定演算子とド・モルガンの法則（練習）

```
/* loop_do_kai.c */
#include <stdio.h>

Int main(void)
{
	int no = 0;

	do{
		no = no + 1;
		printf(“%d¥n”, no);

	}while(!(no >= 100));

	return 0;
}
```

2． do文

- 論理否定演算子とド・モルガンの法則（練習）

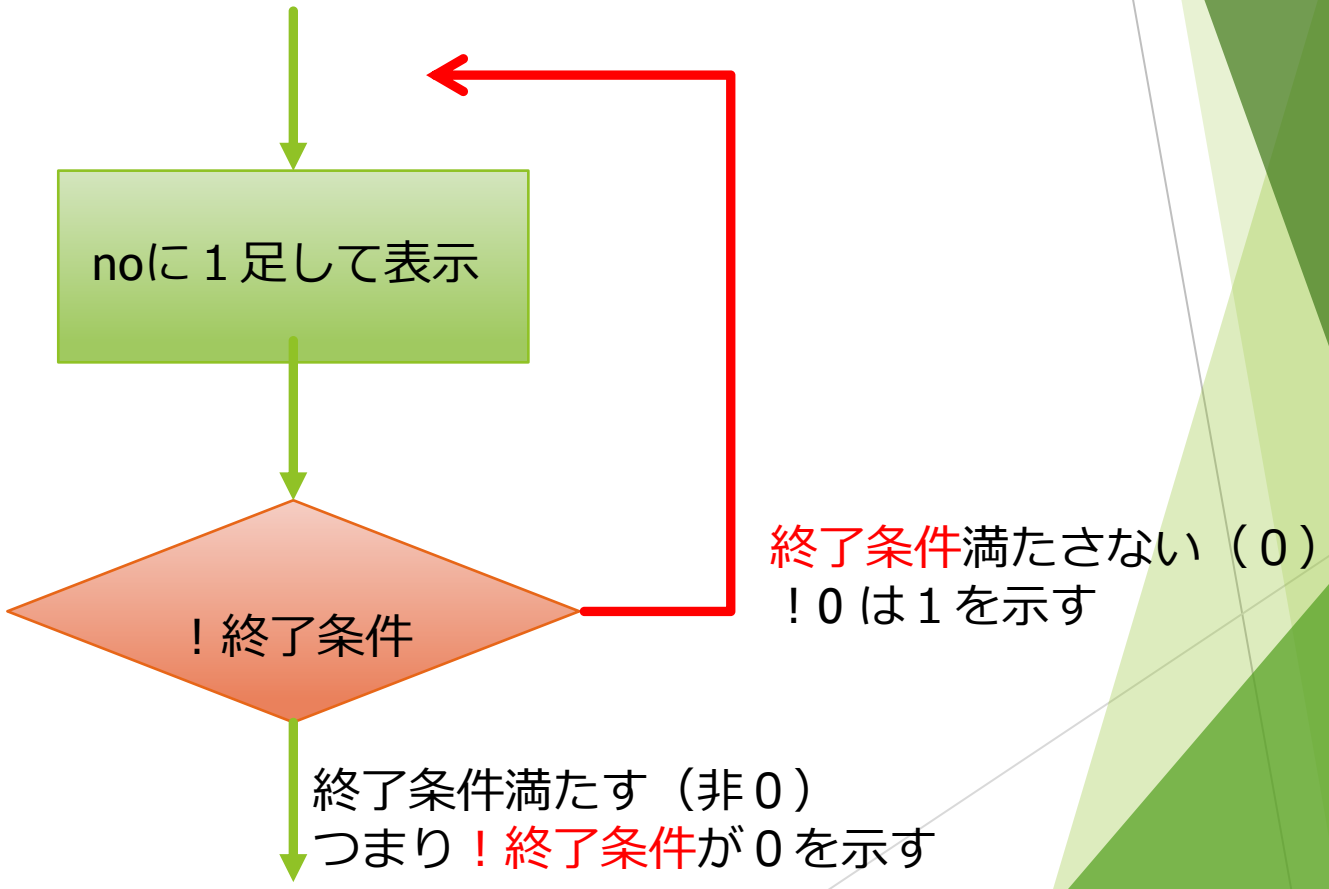
・実行結果



2． do文

- 論理否定演算子とド・モルガンの法則

```
/* loop_do_kai.c */
#include <stdio.h>

Int main(void)
{
        int no = 0;

        do{
                no = no + 1;
                printf(“%d¥n”, no);

        }while(!(no >= 100));

        return 0;
}
```



2． do文

- インクリメント（増分）・デクリメント（減分）

・値を１だけ増やす,もしくは減らす時に使う。

・プログラムを簡潔に書ける

例）

no = no + 1;　  　no++;

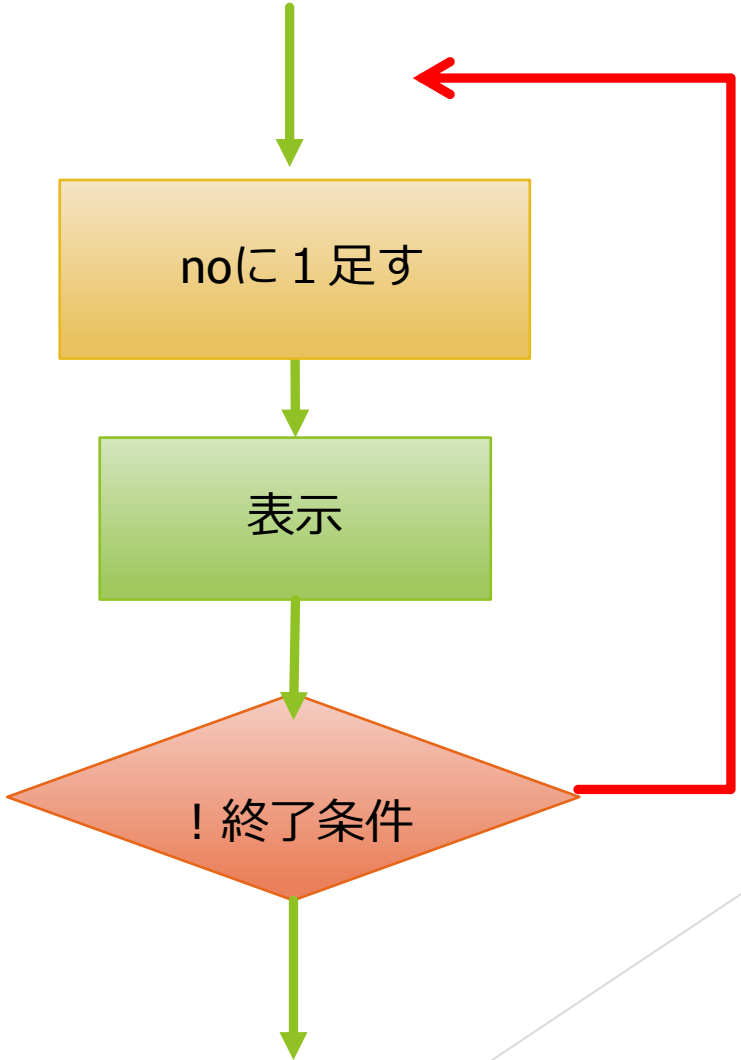
演算子名	書き方	説明	式全体を評価すると…
後置増分演算子	a++	aの値を１増やす	増分前の値
後置減分演算子	a--	aの値を１減らす	減分前の値

演算子名	書き方	説明	式全体を評価すると…
前置増分演算子	++a	aの値を１増やす	増分後の値
前置減分演算子	--a	aの値を１減らす	減分後の値

2． do文

- インクリメント（増分）・デクリメント（減分）

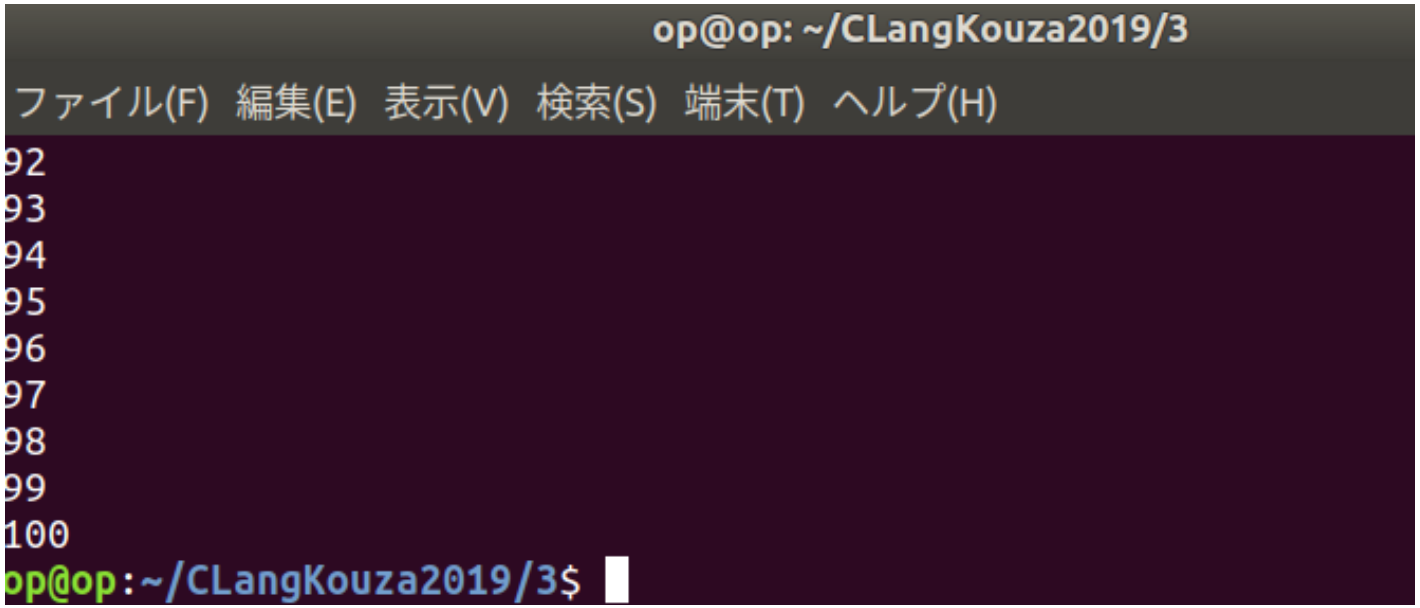
```
/* loop_do_kai.c */
#include <stdio.h>

Int main(void)
{
        int no = 0;

        do{
                printf(“%d¥n”, ++no);

        }while(!(no >= 100));

        return 0;
}
```



2． do文

- インクリメント（増分）・デクリメント（減分）

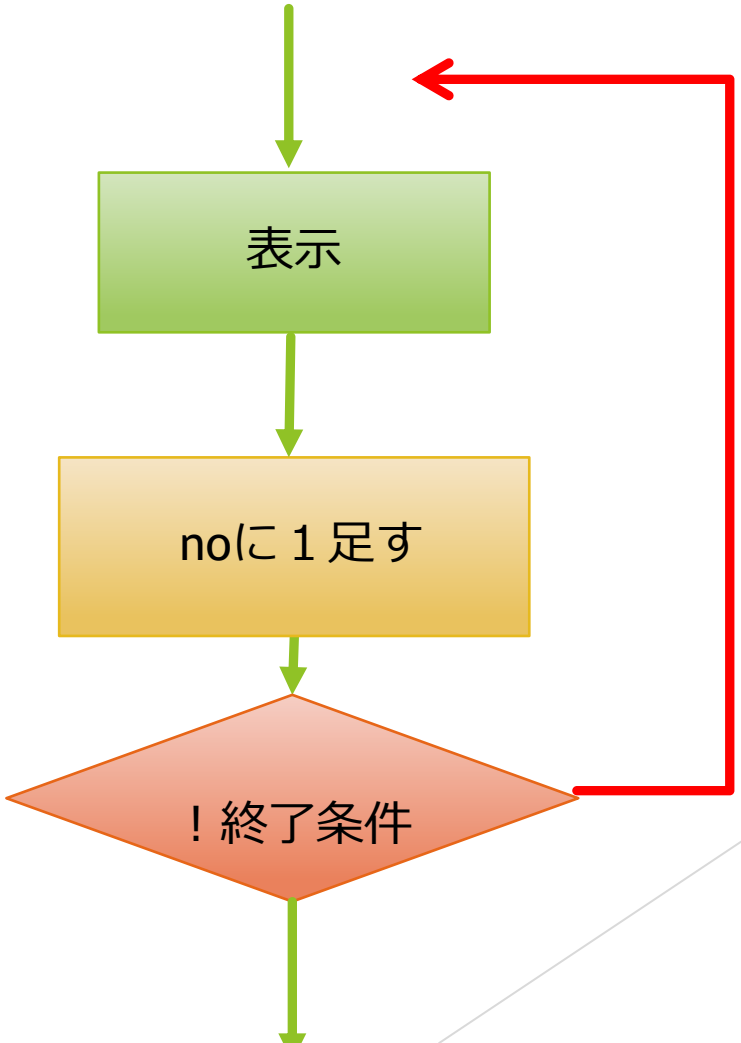
・実行結果



2． do文

- インクリメント（増分）・デクリメント（減分）

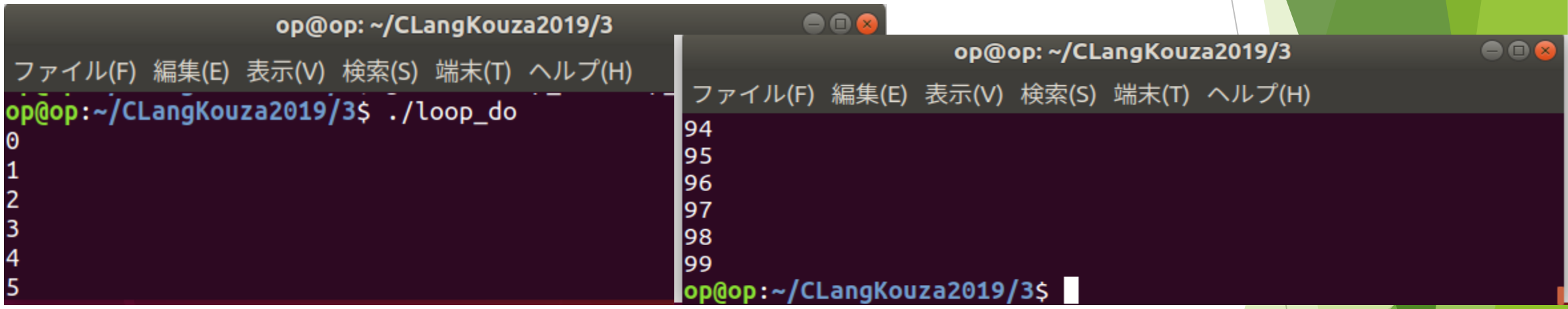
```
/* loop_do_kai.c */
#include <stdio.h>

Int main(void)
{
        int no = 0;

        do{
                printf(“%d¥n”, no++);

        }while(!(no >= 100));

        return 0;
}
```



2． do文

- インクリメント（増分）・デクリメント（減分）

・実行結果



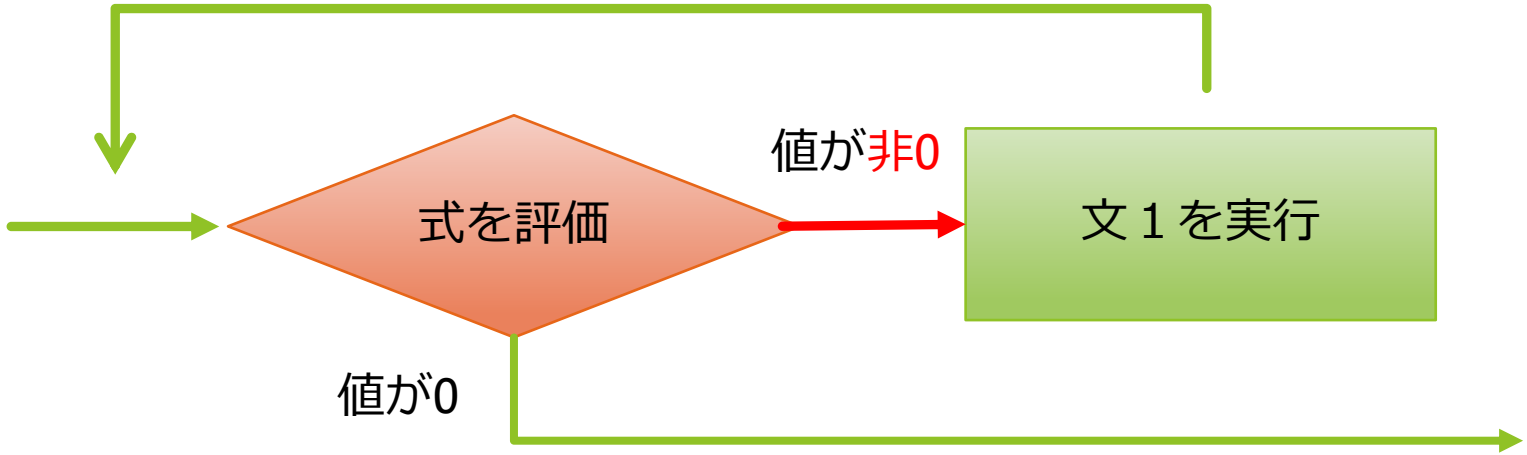
前置と後置どちらを使うかはよく考える！！

目次

3． while文

- while文の書き方

while(式)　文１



・まず式を評価する。その値が非０なら文１を実行する。
実行後再び式を評価する。

do文と違って先に式の評価をする！！
場合によっては一度も実行されない…

3． do文

- while文の書き方

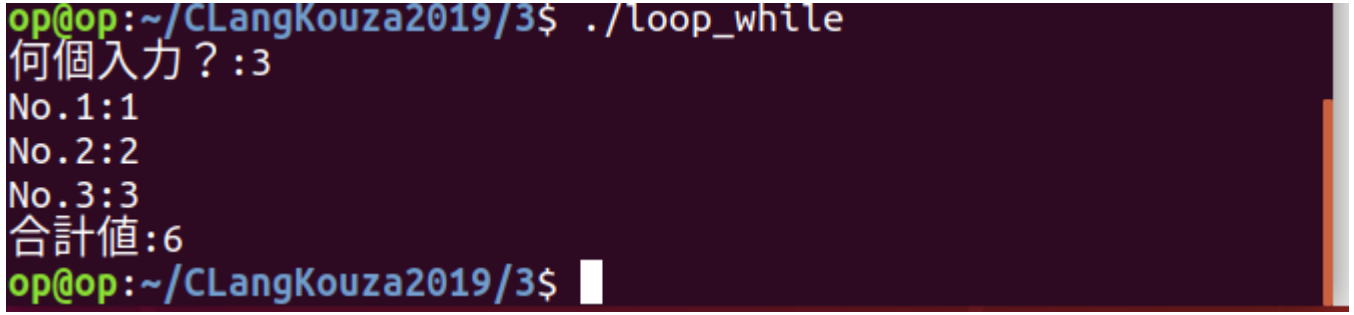
練習：
例）指示された個数だけ整数を読みこみ合計を表示する

```
/* loop_while.c */
#include <stdio.h>

Int main(void)
{
        int i = 0;
        int sum = 0;
        int num ,tmp;

        printf(“何個入力？:”);
        scanf(“%d”, &num);

        while( i < num){
                printf(“No.%d:”, ++i);
                scanf(“%d”, &tmp);
                sum += tmp;
        }

        printf(“合計値:%d ¥n”, sum);

        return 0;
}
```

3． do文

- while文の書き方

練習：
実行例）



3． do文

- while文の書き方

解説：
例）指示された個数だけ整数を読みこみ合計を表示する

```
/* loop_while.c */
#include <stdio.h>

Int main(void)
{
        int i = 0;
        int sum = 0;
        int num ,tmp;

        printf(“何個入力？:”);
        scanf(“%d”, &num);

        while( i < num){
                printf(“No.%d:”, ++i);
                scanf(“%d”, &tmp);
                sum += tmp;
        }

        printf(“合計値:%d ¥n”, sum);

        return 0;
}
```

この“+=”を複合代入演算子という

以下の２つはほぼ同じ

```
sum =  sum + tmp;

sum += tmp;
```

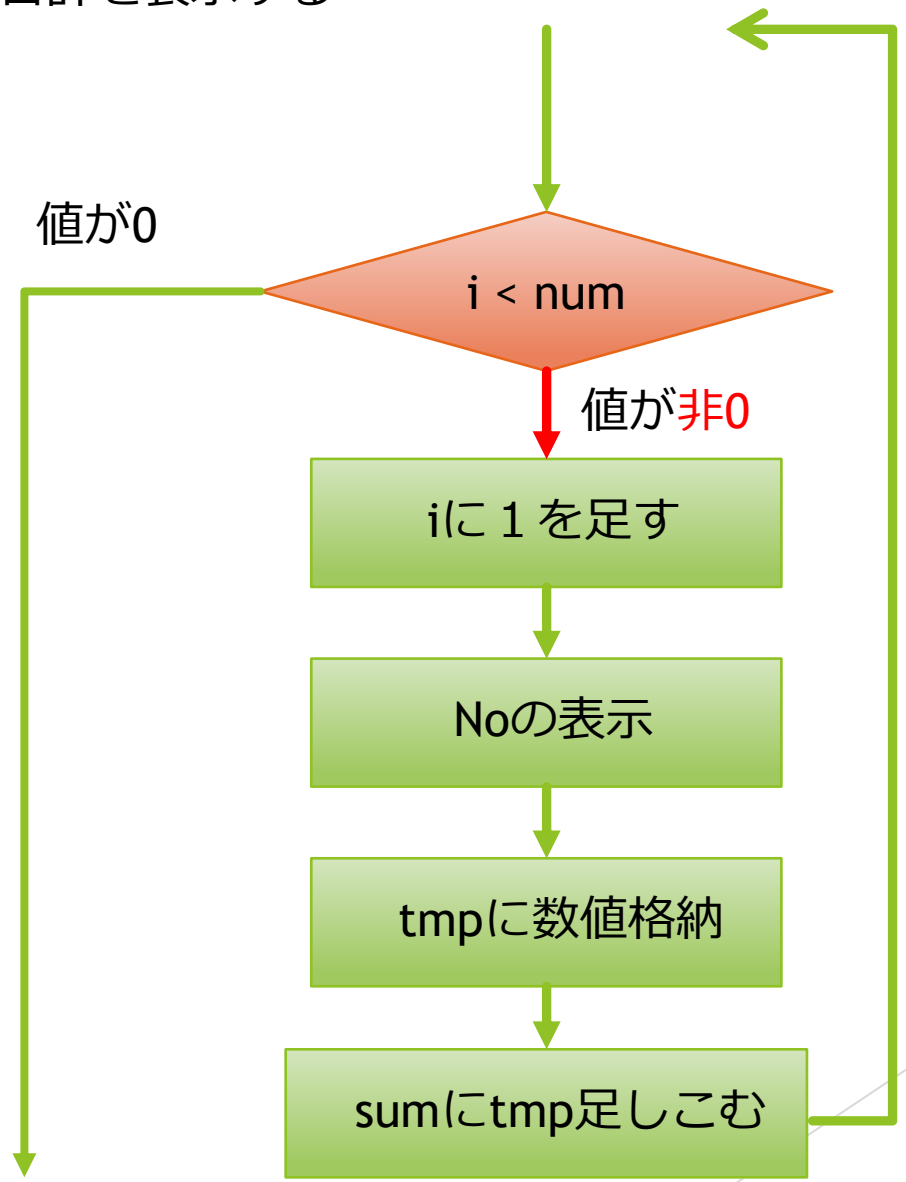
複合演算子の他の例

-=, *=, /=, %=

など

解説：
例）指示された個数だけ整数を読みこみ合計を表示する

```
/* loop_while.c */
#include <stdio.h>

Int main(void)
{
        int i = 0;
        int sum = 0;
        int num ,tmp;

        printf(“何個入力？:”);
        scanf(“%d”, &num);

        while( i < num){
              printf(“No.%d:”, ++i);
              scanf(“%d”, &tmp);
              sum += tmp;
        }
        printf(“合計値:%d ¥n”, sum);

        return 0;
}
```

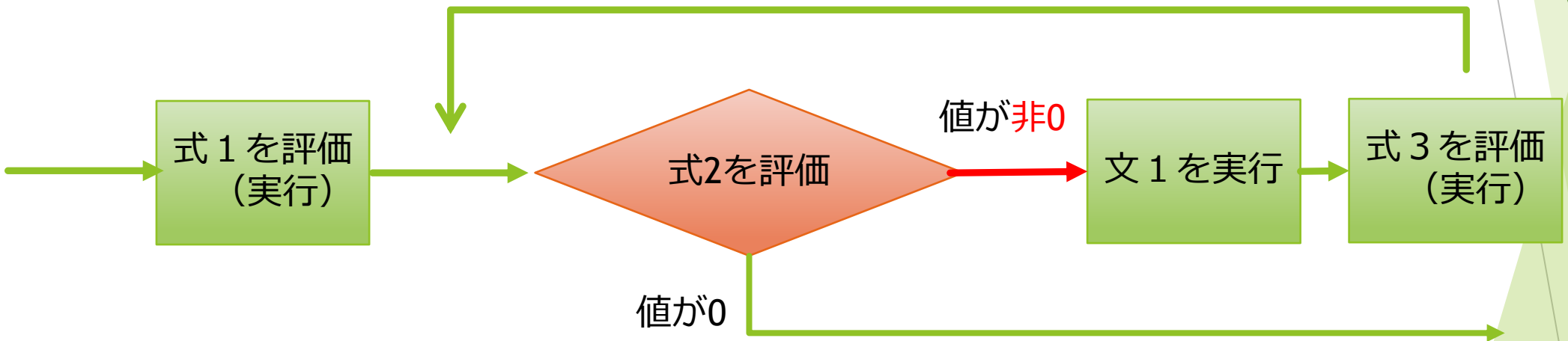


目次

４．　for文

- for文の書き方

for(式1; 式2; 式3)　文１



・式１：前処理。繰り返し前に１度だけ実行される。必要ないなら省略。

・式２：繰り返しを継続するか判定。省略すると無限ループになる。

・式３：後始末。ループ本体(文１)の実行後に実行される。
　　必要ないなら省略。

４．　for文

- for文の書き方(練習)

例)
入力してもらった数の大きさ分＊を表示する。

```
/* loop_for.c */
#include <stdio.h>

Int main(void)
{
        int i , no;

        printf(“何個？:”);
        scanf(“%d”, &no);

        for( i=0; i<no; i++)
                putchar(‘＊’);

        putchar(‘¥n’);

        return 0;
}
```

４．　for文

- for文の書き方(練習)

実行例）

４．　for文

- for文の書き方(練習)

解説）

```
/* loop_for.c */
#include <stdio.h>

Int main(void)
{
        int i , no;

        printf(“何個？:”);
        scanf(“%d”, &no);

        for( i=0; i<no; i++)
                putchar(‘＊’);

        putchar(‘¥n’);

        return 0;
}
```

putchar関数：
　文字を単一引用符‘で囲んだもの（文字定数）を表示する。

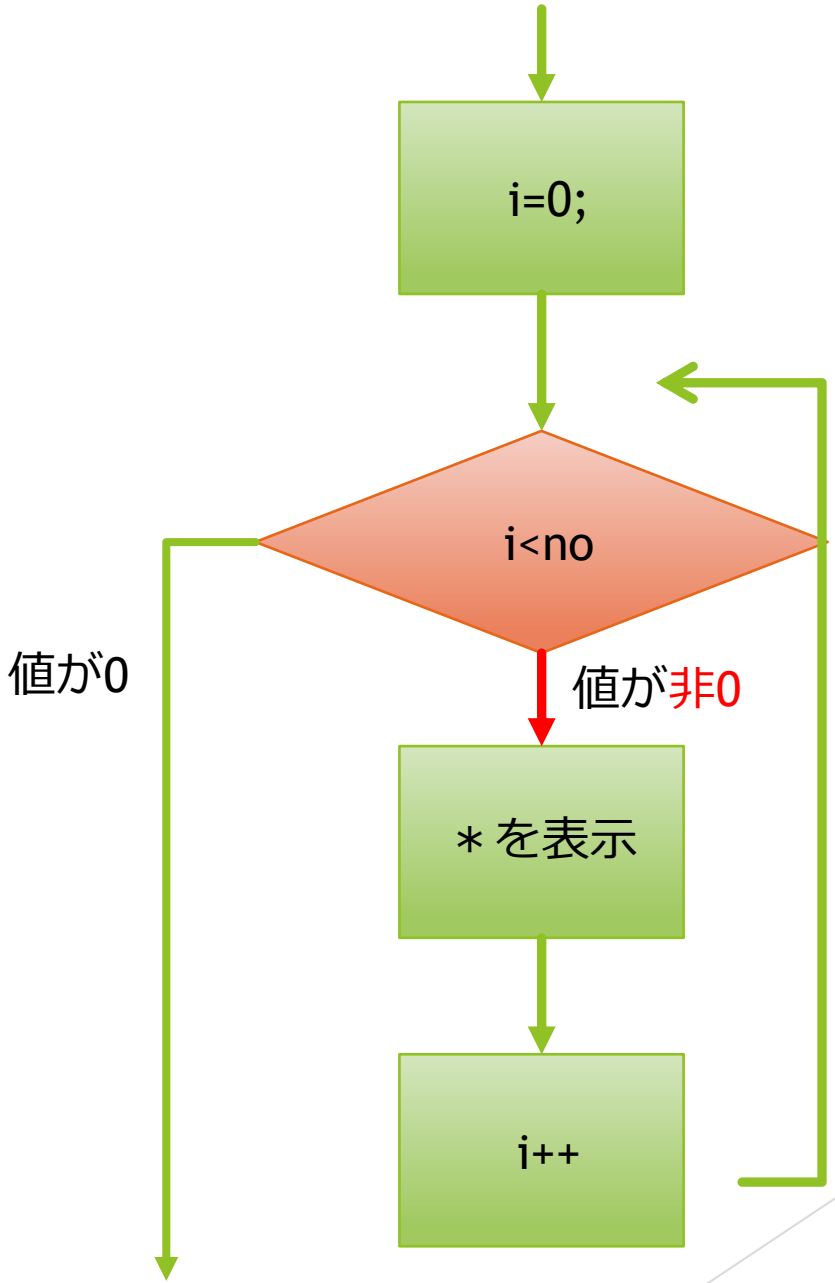
ABCは文字列なので‘ABC’は誤り！！

４．　for文

- for文の書き方(練習)

解説）

```
/* loop_for.c */
#include <stdio.h>

Int main(void)
{
        int i , no;

        printf(“何個？:”);
        scanf(“%d”, &no);

        for( i=0; i<no; i++)
                putchar(‘＊’);

        putchar(‘¥n’);

        return 0;
}
```



目次

５．　多重ループ

- 多重ループとは？

繰り返しの中で繰り返しを行うこと。

例）

```
for{
    for{
        while(・・・){
            ・・・
        }
    }
}
```

forの中でdoやwhileを使っても多重ループ

5． 多重ループ

- 多重ループ（練習）

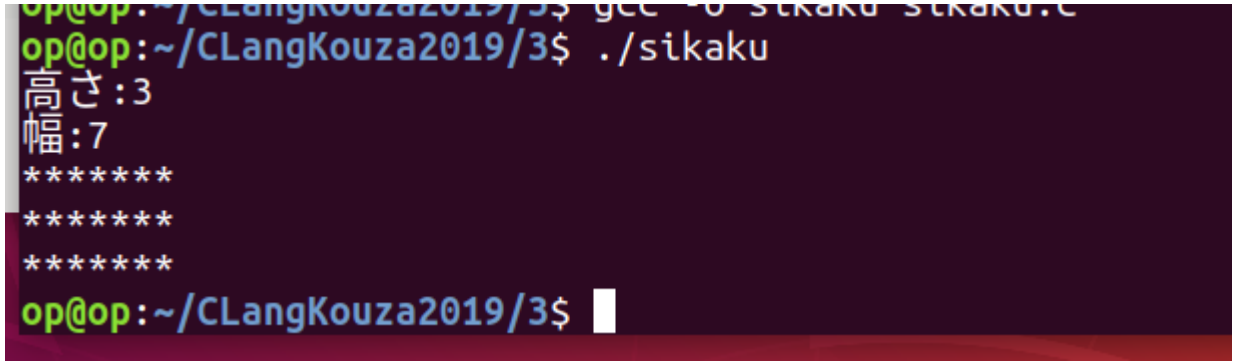
例）
任意の辺の長さの長方形の描画

```
/* sikaku.c */
#include <stdio.h>

Int main(void)
{
        int  i , j;
        int height, width;

        printf(“高さ:”);
        scanf(“%d”, & height);
        printf(“横幅:”);
        scanf(“%d”, & width);

        for( i=1; i<= height; i++)  {
                for(j=1; j<= width; j++){
                        putchar(‘＊’);
                }
                putchar(‘¥n’);
        }

        return 0;
}
```

5． 多重ループ

- 多重ループ（練習）

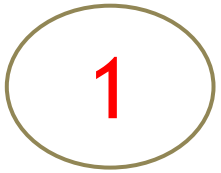
実行例）
　任意の辺の長さの長方形の描画



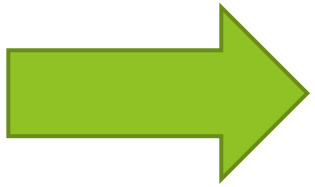
```
/* sikaku.c */
#include <stdio.h>

Int main(void)
{
      int  i , j;
      int height, width;

      printf(“高さ:”);
      scanf(“%d”, & height);
      printf(“横幅:”);
      scanf(“%d”, & width);

     for( i=1; i<= height; i++) {
          for(j=1; j<= width; j++){
               putchar(‘＊’);
          }
          putchar(‘¥n’);
     }

     return 0;
}
```

```
/* sikaku.c */
#include <stdio.h>

Int main(void)
{
      int  i , j;
      int height, width;

      printf(“高さ:”);
      scanf(“%d”, & height);
      printf(“横幅:”);
      scanf(“%d”, & width);

     for( i=1; i<= height; i++) {
          for(j=1; j<= width; j++){
               putchar(‘＊’);
          }
          putchar(‘¥n’);
     }

     return 0;
}
```

i　　j

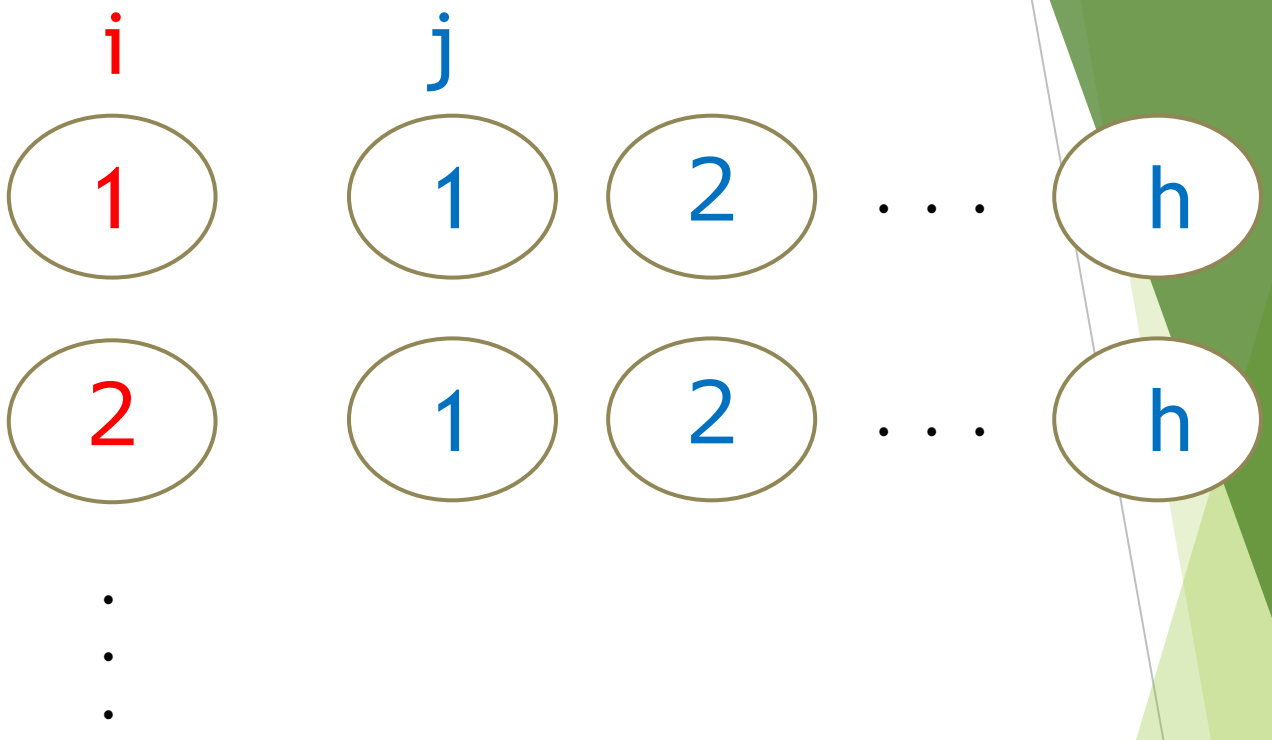


```
/* sikaku.c */
#include <stdio.h>

Int main(void)
{
      int  i , j;
      int height, width;

      printf(“高さ:”);
      scanf(“%d”, & height);
      printf(“横幅:”);
      scanf(“%d”, & width);

     for( i=1; i<= height; i++)  {
          for(j=1; j<= width; j++){
               putchar(‘＊’);
          }
          putchar(‘¥n’);
     }

     return 0;
}
```



この段階で１段目の＊が出来上がる

```
/* sikaku.c */
#include <stdio.h>

Int main(void)
{
      int  i , j;
      int height, width;

      printf(“高さ:”);
      scanf(“%d”, & height);
      printf(“横幅:”);
      scanf(“%d”, & width);

     for( i=1; i<= height; i++) {
          for(j=1; j<= width; j++){
               putchar(‘＊’);
          }
          putchar(‘¥n’);
     }

     return 0;
}
```

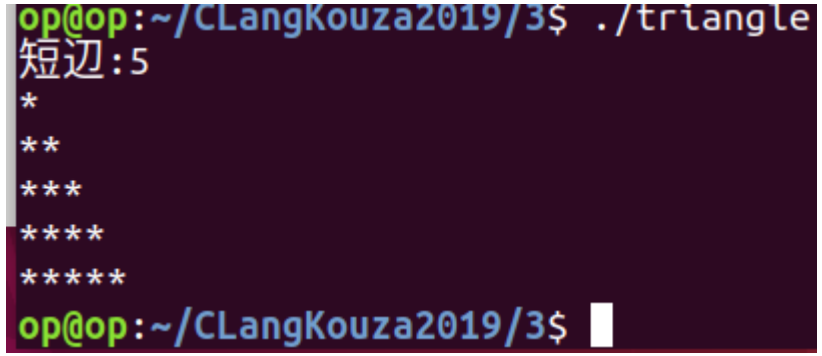


i<= heightを満たさなくなる,つまり
入力した高さになるまで繰り返す。

5． 多重ループ

- 多重ループ（演習）

問）
　入力してもらった短辺の長さを持つ左下直角三角形を描画するプログラムを書け



5． 多重ループ

- 多重ループ（演習）

解答）

```
#include <stdio.h>

int main(void)
{
	int i, j;
	int len;

	printf("短辺:");
	scanf("%d", &len);

	for(i=1; i <= len; i++){
		for(j=1; j<= i; j++){
			putchar(‘＊');
		}
		putchar('¥n');
	}

	return 0;
}
```

今回の内容は以上です。